Development of an Aerial Robot for Flying in Confined Spaces and Interacting with Ground Robots

Lukas Vacek
Penn Aerial Robotics – University of Pennsylvania

Benjamin T. Kramer
Penn Aerial Robotics – University of Pennsylvania

ABSTRACT

Penn Aerial Robotics will present a solution to Mission 7 of the International Aerial Robotics Competition 2016. In this competition, a single quadcopter will herd 10 ground robots towards a goal line while avoiding moving obstacles. Our solution actively simulates the position of the ground robots to determine which ground robots are at risk of going out of bounds and require a direction change. Accurate maneuvering, navigation and ground robot tracking is achieved through computer vision using downward and front facing cameras. The Intel NUC on board the quadcopter collects data from various data sources through ROS and runs advanced algorithms, including optical flow, feature detection and inertial navigation to feed accurate vision position estimates to the onboard autopilot. Control of the quadcopter is achieved by sending set points through MAVLink to the Pixhawk, which then executes the necessary maneuvers to reach the desired 3D position and orientation.

INTRODUCTION

Statement of the Problem

In Mission 7 of the International Aerial Robotics Competition (IARC), there are 10 ground robots and 4 moving obstacles of various heights inside a 20 meter by 20 meter playing field. The aim of the mission is to herd all 10 ground robots across the green goal line in the playing field. The playing field is situated indoors, hence does not have any GPS signal, and features gridlines spaced 1 meter apart to ease navigation.[i]

Conceptual Solution to Solve the Problem

The quadcopter selected for this project uses high-performance off-the-shelf components for the airframe and power system. The autopilot used is the Pixhawk with the PX4 firmware. This was chosen for its excellent development support, extensive functionality and well-tested codebase.

Given the large amounts of processing power needed to analyze the images obtained from the two onboard cameras, we decided to use processing power onboard the quadcopter for all the control systems and algorithms necessary for the mission. To achieve this, we added an Intel NUC

computer to the quadcopter and use the ground station only to monitor the status of the flight systems. The NUC runs the Robot Operating System (ROS), which provides excellent versatility for managing various data streams and outputs. Our main IARC control application runs as a ROS node and communicates with the Pixhawk through the MAVROS node, which uses the MAVLink protocol to transmit messages to the Pixhawk.[ii]

The ground station can monitor the status of the entire system both through the long-range 2.4GHz Ubiquiti airMAX and also through the 915 MHz telemetry radios. The airMAX connects the Intel NUC to the ground station through the local network, and the ground station can also act as an Internet gateway. This provides excellent versatility as team members can SSH into the quadcopter to monitor system status, run programs and pull latest code updates. The telemetry radios connect the Pixhawk MAVLink data streams directly to QGroundControl running on the ground station laptop.

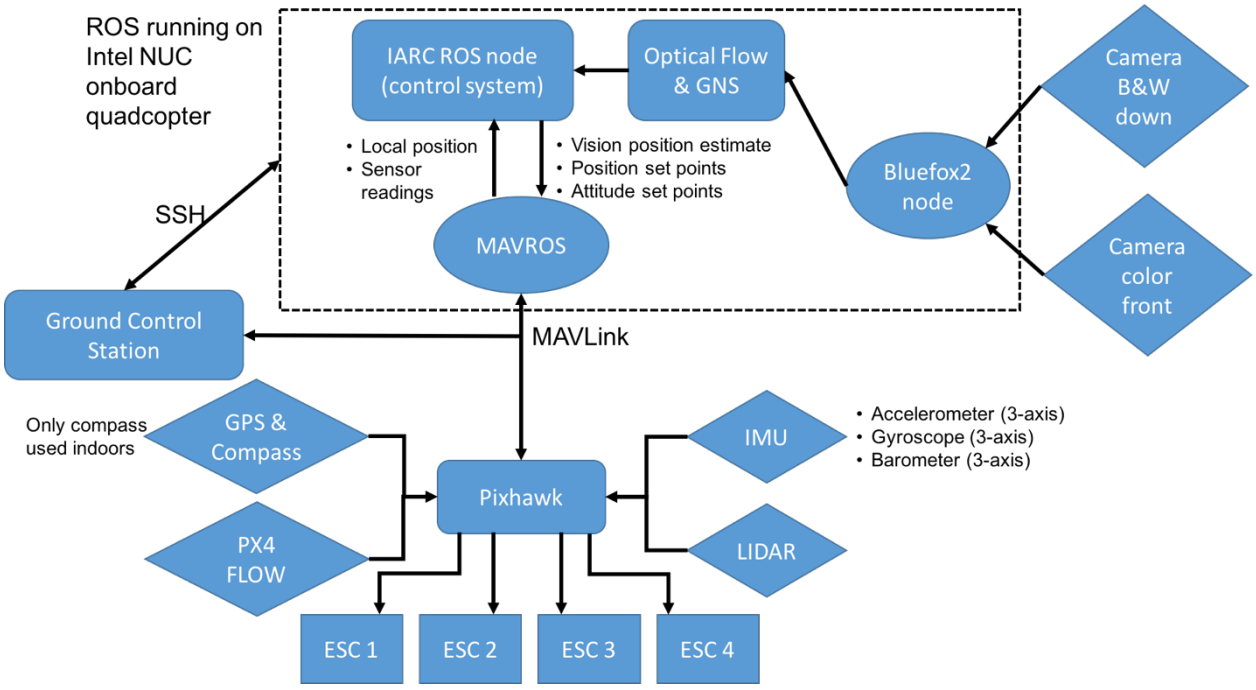

Figure 1: Overall System Architecture

Note: The quadcopter does have a GPS installed as the Pixhawk GPS receiver and compass are installed on the same sensor module. However, only the compass is used as the GPS does not have signal indoors.

Yearly Milestones

Spring 2014	Penn Aerial Robotics is created as a club at the University of Pennsylvania
September 2015	Club decides to officially pursue IARC competition
November 2015	Quadcopter design finalized and assembled
December 2015	First flight test with manual control and Pixhawk autopilot
January 2016	First flight test with offboard control through MAVROS and ROS
March 2016	Begin work on indoor navigation and ground robot simulation
April 2016	First successful tests of optical flow feature recognition
May 2016	Successful test of precise, indoor position hold using optical flow and LIDAR
June 2016	Started integration testing with ground robots

AIR VEHICLE

Figure 2: The quadcopter used in the competition

Propulsion and Lift System

The quadcopter used for this competition is based on the DJI F450 frame and DJI E310 propulsion set. DJI E420S electronic speed controllers are used, with four 2312 motors and 9-inch, self-tightening Z-blade propellers (model number 9450). This is powered by a 4S 6600 mAh 10C MultiStar battery. The battery was chosen for its high-capacity; while this means that the discharge rate (10C) is lower than most LiPos, when multiplied by the large capacity (10C × 6.6Ah = 66A max current) the maximum current the LiPo can handle is still well above the maximum current all the electronics will draw (~50A from testing). Therefore, it was the best choice as it has the highest capacity per weight of all batteries of comparable weight.

This airframe, propulsion and lift system was primarily chosen for its durability, power and maneuverability, as well as ease of assembly and maintenance. The system provides 8000N of static force at 12V and approximately 9500N at 14.8V. It provides sufficient power to lift and maneuver the quadcopter, weighing 2.0 kg. Furthermore, the E310 propulsion system also features active braking, which provides much more accurate control of the quadcopter. Performance testing showed that the quadcopter could perform aggressive maneuvers of up to 60° banks (maximum limit we deemed safe to test) and could reach a top speed of 53 mph.

Guidance, Navigation and Control

Top-level guidance, navigation, and control is performed on a Core i5 Intel NUC running ROS. Code is normally written in Python; performance-critical code, such as that required for custom image processing, is written in C++. A collection of nodes performs all the necessary tasks required to execute the mission.

The guidance node is responsible for making step-by-step decisions that will lead to ultimate mission success. Our algorithm functions off the belief that the best way to complete Mission 7a fully is to herd the ground robots across the green line as quickly as possible in the 10-minute period. This involves remaining within bounds as long as possible and implementing a herding strategy to keep all ground robots within the arena, and then proceeding with choosing which ground robots to direct across the green line. To this end, our algorithm makes decisions according to the following rough priority structure:

1. Avoid collisions
2. Keep ground robots within bounds
3. Score ground robots across the green line

Collision avoidance is the top priority of our algorithm for 2 reasons. First and foremost, our vehicle does not have a protective frame around it, so it is highly likely that any collision we make will lead to severe, mission-comprising damage to the propellers or vital electronics. Second, given that we are allowed only 2 collisions, and that unexpected issues may arise in the arena or with our control software, it is of utmost importance for us to take care not to incur a disqualification for failure to avoid obstacles.

Our collision avoidance strategy therefore seeks to minimize any chance of making contact with the obstacles. Our vehicle attempts to fly at or near the allowed 3m limit for most of the flight, so that we can safely fly over obstacles for most of the time. Other decisions regarding herding of the ground robots, scoring, and path planning are guided by our knowledge of our current proximity to the obstacles. Our path planning algorithms therefore make use of the maximum allowed flying area. Our vehicle will exit the bounds of the ring up to the allowed distance limit for up to 4 seconds in order to attempt to guide itself safely around the obstacles. Special care is taken when obstacles are close to the ground robots we are currently trying to interact with. Our vehicles will make aggressive, vertical descents not only in the interest of saving time but also to minimize time spent near the obstacles. When determining a path towards a ground robot, the vehicle will try to find a path that does not intersect with the projected path of the obstacle. In the event that a ground robot in danger of exiting the bounds is within a "danger zone" (within a certain tunable distance to an

obstacle), and only paths with an unacceptable chance of danger are found, the vehicle will opt to allow the ground robot to exit the field.

The second priority, keeping ground robots within bounds, is met using a "herding" strategy that attempts to steer all ground robots towards the center of the ring, so that the ground robots can spend as much time as possible safely traversing the arena, freeing up time to spend scoring ground robots.

We therefore use a path-planning strategy that takes as input:

1. The current position and velocity of the quadcopter
2. Position and velocity estimates of the ground robots
3. Position and velocity estimates of the obstacles.

The path planner first evaluates a cost function for each ground robot. The cost associated with a ground robot is a function of its projected time to cross a white or red boundary or its distance to the green line, the type of landing strategy required to correct the path of ground robots leaving the field, and its distance from the vehicle and its proximity to an obstacle. Robots closer to the white/red boundaries of the field and further from the vehicle have higher costs, and robots closer to the green boundaries of the field have lower costs. Also associated with high-cost ground robots is a landing strategy (land in front of/on top of) These costs are input into a graph. Ground robots in close proximity to an obstacle and exiting the boundary are deleted from the graph.

A version of Djikstra's algorithm is run on the graph, producing a general path to be run. Actual traversal of the edges of the graph are adjusted according to the location of ground obstacles and the required landing strategy. The vehicle then issues control instructions in 10ms time steps, and the graph is updated every 300ms to account for changes in ground robot motion. Actual trajectories consist of bang-bang trajectories towards local set points along the path to a target.

Control commands from the guidance node are outputed to a MAVROS node communicating with the Pixhawk that performs actual control of the vehicle. The Pixhawk is fed velocity, rotation, and altitude commands at the rate of once every millisecond. In turn, the Pixhawk computes the required logic level output to each of the ESCs. The ESCs onboard communicate with one motor each, issuing the proper applied voltage.

Stability Augmentation System

In order to ensure safe operation during flight, the guidance system will detect when the quadcopter engages in "dangerous" maneuvers, i.e. when the quadcopter attempts to pitch or roll at an angle above 45 degrees. It then proceeds to both decrease velocity and recompute a safer trajectory towards its target.

The Pixhawk is also responsible for stabilization of the quadcopter; a highly tuned pitch and roll controller onboard helps ensure that the vehicle does not perform "dangerous" maneuvers.

Navigation

Navigation is performed by another ROS node responsible for acquiring vehicle, ground robot, and obstacle state. Velocities and positions are determined relative to the center of the arena. Vehicle pose and velocity is estimated using Kalman-filtered data from an IMU onboard the 3DR Pixhawk, and position and velocity are corroborated using a custom optical flow algorithm combining the advantages of the Lucas-Kanade and Horn-Schunck methods[iii] and utilizing OpenCV functions for image acquisition and transformation and Eigen for custom-written methods requiring matrix operations. The output of the optical flow algorithm is used in conjunction with the 1D Lidar and pitch information in a projective transformation algorithm to compute an estimate of ground speed.

Ground robot and obstacle positions are computed using custom image tracking methods run every 500ms. Optical flow is also used as the basis of an algorithm computing the flow of the points comprising the ground robots and obstacles, and the velocity of the quadcopter is subtracted from these velocities to get an estimate of ground robot and obstacle velocity. From these velocities and acquired position, ground robot and obstacle positions are estimated every 100ms.

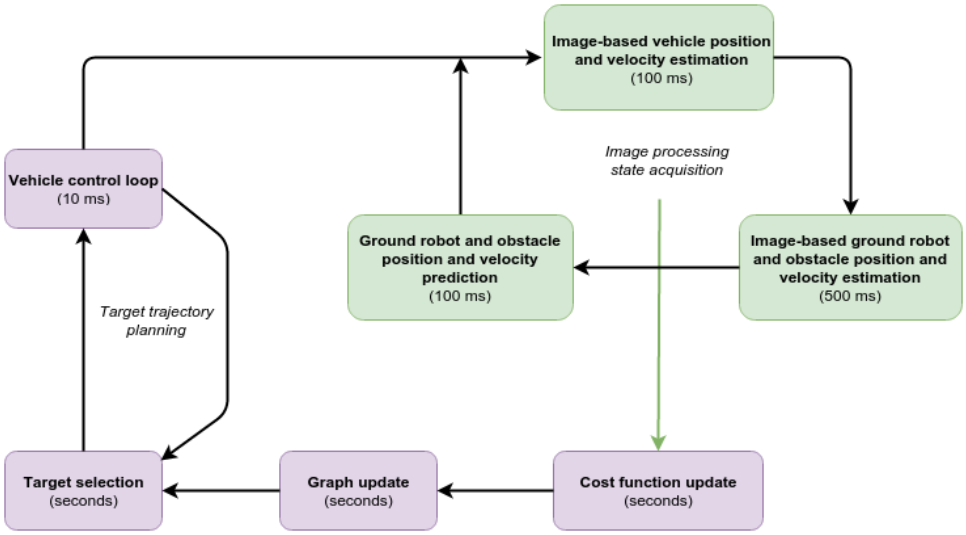

Figure 3: Control System Architecture

Flight Termination System

We identified the following 6 termination conditions and corresponding actions:

1. 10 minutes have elapsed since mission start: the vehicle detects the time elapsed and begins a controlled landing.
2. All ground robots exit the arena: the vehicle detects this condition and begins a controlled landing.
3. Quadcopter illegally exits arena for over 5 seconds: vehicle detects illegal position and time elapsed in 2m buffer space around arena and holds altitude; human operators perform controlled landing.
4. Mission stopped due to ground robot failure: human operators make controlled landing.

5. Control system failure: quadcopter or human operator makes controlled descent, depending on severity of failure.
6. Emergency failure: remote kill switch is activated, cutting power to the vehicle.

PAYLOAD

Sensor Suite

TABLE 1: SENSORS ON THE QUADCOPTER THAT ARE USED FOR THE IARC MISSION

Type	Sensor	Purpose
Camera	Matrix Bluefox2 Color	Object detection and avoidance
	Matrix Bluefox2 B&W	Grid navigation and robot detection
Optical Flow	PX4FLOW	Accurate position hold
	LIDAR	Accurate ground distance
Pixhawk IMU[1]	Accelerometer (3 axis)	Flight stabilization, orientation and attitude control
	Gyroscope (3 axis)	
	Barometer	Altitude (LIDAR has precedence)
External compass	Magnetometer (3 axis)	Orientation

GNC Sensors

The GNC sensors consist of the IMU (accelerometer, gyroscope and barometer) on the Pixhawk, the compass, LIDAR and PX4FLOW sensor. The Pixhawk by itself is capable of accepting data from all these sensors and using it for maneuvering the quad. Our top-level ROS node controlling the quadcopter and mission is additionally capable of tapping into these sensors and controlling the quadcopter with higher precision (e.g. to 'tap' a robot). The PX4FLOW provides positional information for short-term durations and is mainly useful for more accurate position holds.

Mission Sensors

The main mission sensors are the forward and downward facing cameras connected through USB to the Intel NUC. The ROS optical navigation node uses imagery from these cameras to feed vision position estimates to the Pixhawk. These are then used by the Pixhawk as a substitute for GPS and provide accurate positional information allowing us to send positional, rather than attitude, set points.

[1] IMU: Inertial Measurement Unit

Target Identification

The imagery from the downward camera is analyzed for target detection and identification of the ground robots and maneuver to touch or block them.

Threat Avoidance

The front-facing camera is used to detect the moving obstacles in the arena and execute obstacle-avoidance maneuvers.

Communications

The quadcopter is equipped with five different radios for communications. The first is the RC link between the human pilot and the quadcopter. This allows the pilot to remain in control of the quadcopter at all times and switch between 'manual' and 'offboard' mode by flipping a switch. In order for the Pixhawk autopilot to communicate with the onboard computer, there is a USB to serial converter plugged in to the Intel NUC and the TELEM2 port on the Pixhawk. This link allows the Pixhawk and onboard computer to easily communicate with each other by sending messages and instructions.

Communications between the quadcopter and ground station are done through two different channels: a 915MHz telemetry link that transmits MAVLink packets between the Pixhawk and ground station computer. The second is the long-distance Ubiquiti airMAX that connects the base station with the onboard computer. This allows us to treat the quadcopter as if it were on the local network, including the ability to connect over SSH and screen sharing (VNC).

Power Management System

The quadcopter is powered by one 4 cell LiPo battery pack, with a nominal voltage of 14.8V and a capacity of 6600 mAh, producing a total of 87.7 kW.[iv] The power is fed from the battery into the Pixhawk module, which powers the Pixhawk autopilot and all of the sensors (except the cameras which are powered from the NUC). This module also measures the voltage and current flowing out of the battery. From this data, past consumption is calculated so we know how much battery is remaining. After this module, the power is distributed to the ESCs through the built-in distribution board on the F450 frame. Next, the power feeds through a 10A circuit breaker to protect expensive electronics from surges and short-circuits and connects to the Intel NUC and Ubiquiti Pico Station.

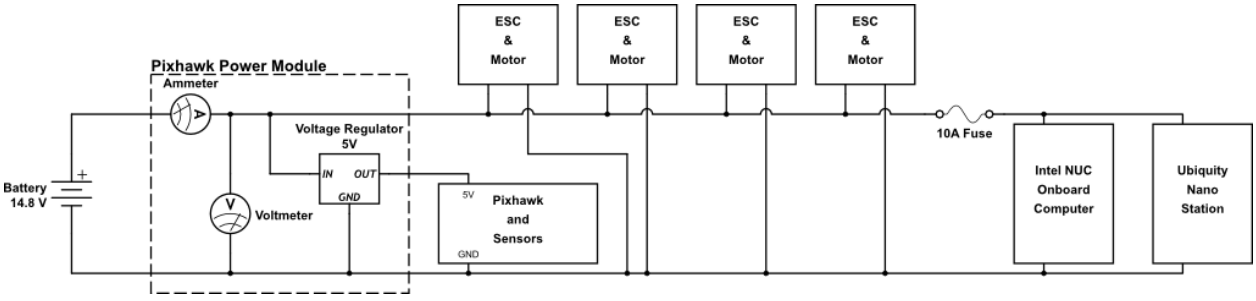

Figure 4: Power distribution system[v]

OPERATIONS

Flight Preparations

Flight preparations are done strictly according to the checklist below to ensure safety, minimize risk of crashes and collisions, and successfully complete the mission. Each flight is conducted under the supervision and control of a safety pilot – the ultimate responsible authority. Equipment is physically checked and replaced as necessary. Software performs internal hardware and system checks. The pilot briefs team members and bystanders on risk management and flight objectives.

Checklist

Initial	Pre-arm
Briefing - pilot briefs objectives and risks **Motors** - secure, spin freely and screws tight **Props** - tight, balanced, no chips or dents **Airframe** - no cracks or damage, all screws and items secured **Electronics** - no loose, frayed wires **Battery** - no damage, charged to 14.8V **Fuse** - not broken **Ground station** - Connect 3DR radio + network **Ground station (QGC)** - ON **Transmitter** - battery charged, MODEL1 selected **Transmitter** - ON CONNECT BATTERY **Kill switch** - check (ABORT) **Reconnect power**	**ESCs** - pulsing red or green lights/no error **Pixhawk** - no error **QGC** - connected **Intel NUC** - ON **SSH** - connect to quadcopter **Launch ROS** (*roslaunch pennair quad.launch*) **Calibrate local position and sensors** (*rosrun pennair calibrate*) **ABORT person** - standby **Area clear!** **Safety switch** - ON ARM **PX4 safety checks** - all passed. Quadcopter armed.

Takeoff	In-flight (continuous)	After landing
Mission - signal READY! **Start IARC Mission program** (*rosrun pennair iarc*) OFFBOARD MODE ON	**Battery** - 20% or more **Quadcopter** - in bounds **Control system** - no errors **Emergency** - ABORT	DISARM **Intel NUC** - shutdown **Disconnect battery** **Props** - no damage **Motors** - no overheat/damage

Man/Machine Interface

The quadcopter is controlled by humans both from a manual RC transmitter and from the ground station. While mission programs are initiated and monitored over SSH and from QGroundControl, the quadcopter must be manually armed and put into 'OFFBOARD' mode manually from the RC transmitter. This allows the human operator to regain control at any time if necessary. There is also an emergency kill switch that cuts all power to the quadcopter, resulting in all motors stopping and an uncontrolled landing.

The operation of the quadcopter ideally requires three individuals: one pilot to man the RC transmitter, one person to monitor the data shown in the ground station and one person to man the kill switch.

RISK REDUCTION AND SAFETY

Risk reduction consists of both onboard and human-interactive measures to ensure safe operation of the vehicle in flight.

The guidance system attempts to take rather safe trajectories, opting largely to perform a series of straight-line, bang-bang trajectories followed by rotations to minimize the chance of performing high-angle maneuvers. The guidance system also imposes a limit on velocity during the mission to 75% of the maximum possible speed. If the quadcopter is flying more than 3m outside of the ring, the guidance system indicates failure and holds position, switching the flight mode to human-controlled altitude mode.

In addition, each packet of data from the IMU, cameras, and LIDAR is monitored to ensure that these components are functional. Data differing by excessive margins between packets or frames is considered indicative of failure of the equipment. In the event of a single component failure, less-reliable methods of position and velocity estimation are used. In the event that the vehicle can no longer determine its position and velocity or the estimate these quantities for the ground robots and obstacles, the vehicle will immediately attempt to make a vertical landing.

The Pixhawk also receives a heartbeat system from MAVROS within ROS. In the event that communication between the NUC and Pixhawk fails, the Pixhawk enters a failsafe mode, whereby an alarm sounds and the Pixhawk enters altitude control mode, awaiting commands from the operator, who then performs a safe landing.

Two members of the team must be watching the vehicle at all times. In the event that unexpected emergencies arise during flight, the team members can immediately switch into manual mode. In the case of total system failure or in the event that the vehicle takes a dangerous path outside the ring, a remote kill switch is triggered to cut off power to the system.

Vehicle Status

Vehicle status is maintained by the Pixhawk and GNC system concerning the items above. Information as to the status of critical onboard sensors (IMU, cameras, LIDAR), the ROS system,

the Pixhawk, and the battery voltage are relayed over SSH and telemetry radios to a laptop monitored by a human operator.

Shock/Vibration Isolation

In order to reduce propeller vibration, the vehicle propellers are balanced via sanding. To minimize the transmission of vibrations to the sensors, the Pixhawk, and the NUC, foam is placed between each component and its mount points.

EMI/RFI Solutions

To reduce EFI, an LC power filter is installed between the battery and components requiring 5-12V. To reduce RFI as well, pairs of wires are twisted and are isolated from each other, i.e. groups of wires are not bundled together but rather run along different areas of the vehicle.

Modeling and Simulation

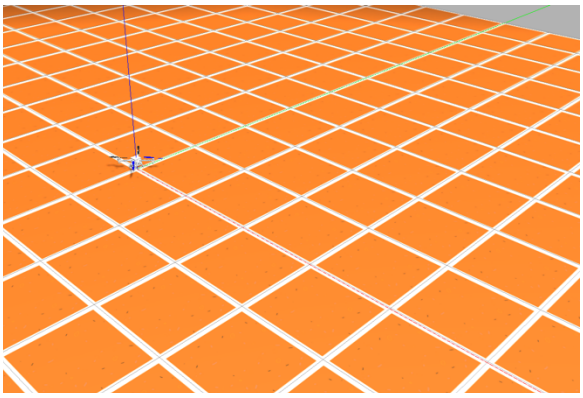

Figure 5: Gazebo simulation of quadcopter on gym floor

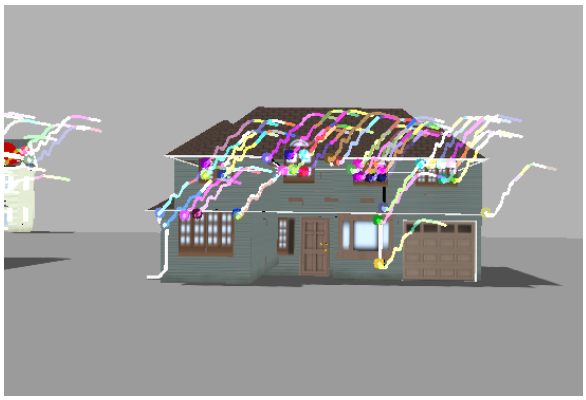

Figure 6: Initial testing of feature recognition and tracking algorithms in Gazebo

The vehicle was simulated in Gazebo. A model of the quadcopter was constructed, and its flight physics were programmed. The simulation interacts with ROS just as the real quadcopter would; it can be issued commands to its Pixhawk, and it provides 6-DOF IMU, front-facing and downward-facing 640x480 camera data, and 1D LIDAR information via Gazebo models and plugins for these sensors. Later Gazebo simulations included the ground robots and obstacles.

Testing

The vehicle was tested at each stage of development, from motor control to initial libraries issuing control commands to the Pixhawk and motors and functions reading data from sensors, to early image processing algorithms, and later to inputs of position and velocity estimates into ROS to perform basic maneuvers such as navigating an area and flying patterns, and finally to guidance algorithms. For tests involving the full chassis, simulation testing was following by testing in a netted 30x30 indoor area.

Integration Testing Pipeline

1. Bench tests of motors and sensors
2. Chassis takeoff/land and area circling test: use IMU and altitude data fed into ROS to control flight using basic libraries in simulation and physical netted area
3. Off-board optical flow velocity/position estimate and feature tracking testing using simulation data
4. Onboard optical flow and feature tracking testing in netted area
5. Vehicle area circling, pattern flying, and waypoint testing using image data in simulation and physical netted area
6. Cost function development and evaluation in Gazebo and implementation of path planner
7. Mission execution in simulation and in netted area using 5 ground robots (2 as obstacles)

CONCLUSION

The design we will showcase features several technical innovations that could provide us with a competitive advantage. In particular, our innovative ground robot simulator constantly estimates the position of the ground robots and allows the quadcopter to decide which one to pursue next and approximately where to locate it. Likewise, our extensive use of computer modelling and simulation allows us to rapidly test many scenarios and tune our algorithms. These innovations combined with our advance GNC system should yield positive results in Mission 7.

AKNOWLEDGEMENTS

We would like to express our gratitude to Dean Vijay Kumar, Dr. Jnaneshwar Das, Professor Daniel Lee and members of the MRSL lab for their extensive and invaluable support, technical advice and help with preparations for this mission. We also wish to thank the University of Pennsylvania's GRASP lab, SAC and ESAC for their support and funding for this competition.

REFERENCES

[i] "Official Rules for the International Aerial Robotics Competition - Mission 7," May 2016, http://aerialroboticscompetition.org/downloads/mission7rules_051016.pdf.

[ii] "PX4 Development Guide," *Pixhawk*, n.d., http://dev.px4.io/index.html.

[iii] Adrés Bruhn, Joachim Weickert, and Christoph Schnörr, "Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods," *International Journal of Computer Vision* 61, no. 3 (2005): 211–31.

[iv] "Multistart High-Capacity Battery 4S 6600mAh," *Hobbyking*, n.d., http://www.hobbyking.com/hobbyking/store/__64442__Multistar_High_Capacity_4S_6600mAh_Multi_Rotor_Lipo_Pack_US_Warehouse_.html.

[v] "3DR Power Module," *Ardupilot*, n.d., http://ardupilot.org/copter/docs/common-3dr-power-module.html.